

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Scott C. Johnson et al.

Filed: Concurrently Herewith

For: NETWORK CONNECTED COMPUTING SYSTEM INCLUDING STORAGE
SYSTEM

Serial No.: Unknown

Prior Group Art Unit: 2152

Examiner: Unknown

Atty. Dkt: SURG:158

NUMBER: EL703719634US

I hereby certify that this paper or fee is being deposited with the United States Postal Service "EXPRESS MAIL POST OFFICE TO ADDRESSEE" service, postage prepaid, under 37 CFR 1.10 on the date indicated below and is address to:
Assistant Commissioner of Patents, Washington, D.C. 20231.

Oliver Ferrara
Signature

5-31-01
Date

Assistant Commissioner For Patents
Washington, D.C. 20231

Dear Sir:

PRELIMINARY AMENDMENT TO REDUCE FILING FEES

Please amend the application as follows.

In the specification:

The rewritten clean versions of all the specification changes are provided below.
Attached at the end of this paper is an Appendix providing an indication of the changes relative to the prior version of the specification, as now required by Rule 121.

On page 1, replace the paragraph from lines 1-4 with:

-- Patent Application for
NETWORK CONNECTED COMPUTING SYSTEM INCLUDING STORAGE SYSTEM

Inventors: Scott C. Johnson, Brian W. Bailey, Rodney S. Canion, Thomas E. Garvens,
Gregory J. Jackson and Roger K. Richter--

On page 1, please replace the paragraph from lines 6-22 with:

This application is a continuation of Application Serial No. 09/797,413, which was filed March 1, 2001 and is entitled "NETWORK CONNECTED COMPUTING SYSTEM", which in turn claims priority from Provisional Application Serial No. 60/187,211, which was filed March 3, 2000 and is entitled "SYSTEM AND APPARATUS FOR INCREASING FILE SERVER BANDWIDTH," from Provisional Application Serial No. 60/246,343, which was filed November 7, 2000 and is entitled "NETWORK CONTENT DELIVERY SYSTEM WITH PEER TO PEER PROCESSING COMPONENTS," from Provisional Application Serial No. 60/246,335, which was filed November 7, 2000 and is entitled "NETWORK SECURITY ACCELERATOR," from Provisional Application Serial No. 60/246,443, which was filed November 7, 2000 and is entitled "METHODS AND SYSTEMS FOR THE ORDER SERIALIZATION OF INFORMATION IN A NETWORK PROCESSING ENVIRONMENT," from Provisional Application Serial No. 60/246,373, which was filed November 7, 2000 and is entitled "INTERPROCESS COMMUNICATIONS WITHIN A NETWORK NODE USING SWITCH FABRIC," from Provisional Application Serial No. 60/246,444, which was filed November 7, 2000 and is entitled "NETWORK TRANSPORT ACCELERATOR," and from Provisional Application Serial No. 60/246,372, which was filed November 7, 2000 and is entitled "SINGLE CHASSIS NETWORK ENDPOINT SYSTEM WITH NETWORK PROCESSOR FOR LOAD BALANCING," the disclosures of each being incorporated herein by reference.

Please replace the paragraph on page 16 from lines 10 -31 and on page 17 from lines 1-3 with the following:

09/797,412

--The processing cores of a network processor are typically accompanied by special purpose cores that perform specific tasks, such as fabric interfacing, table lookup, queue management, and buffer management. Network processors typically have their memory management optimized for data movement, and have multiple I/O and memory buses. The programming capability of network processors permit them to be programmed for a variety of tasks, such as load balancing, network protocol processing, network security policies, and QoS/CoS support. These tasks can be tasks that would otherwise be performed by another processor. For example, TCP/IP processing may be performed by a network processor at the front end of an endpoint system. Another type of processing that could be offloaded is execution of network security policies or protocols. A network processor could also be used for load balancing. Network processors used in this manner can be referred to as "network accelerators" because their front end "look ahead" processing can vastly increase network response speeds. Network processors perform look ahead processing by operating at the front end of the network endpoint to process network packets in order to reduce the workload placed upon the remaining endpoint resources. Various uses of network accelerators are described in the following concurrently filed U.S. patent applications: no. 09/797,412, entitled "Network Transport Accelerator," by Bailey et. al; no. 09/797,507, entitled "Single Chassis Network Endpoint System With Network Processor For Load Balancing," by Richter et. al; and no. 09/797,411, entitled "Network Security Accelerator," by Canon et. al; the disclosures of which are all incorporated herein by reference. When utilizing network processors in an endpoint environment it may be advantageous to utilize techniques for order serialization of information, such as for example, as disclosed in concurrently filed U.S. patent application no. 09/797,197, entitled "Methods and Systems For The Order Serialization Of Information In A Network Processing Environment," by Richter et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 23 from lines 12 - 25 with the following:

--Storage management engine 1040 may employ any suitable method for caching data, including simple computational caching algorithms such as random removal (RR), first-in first-out (FIFO), predictive read-ahead, over buffering, etc. algorithms. Other suitable caching

algorithms include those that consider one or more factors in the manipulation of content stored within the cache memory, or which employ multi-level ordering, key based ordering or function based calculation for replacement. In one embodiment, storage management engine may implement a layered multiple LRU (LMLRU) algorithm that uses an integrated block/buffer management structure including at least two layers of a configurable number of multiple LRU queues and a two-dimensional positioning algorithm for data blocks in the memory to reflect the relative priorities of a data block in the memory in terms of both recency and frequency. Such a caching algorithm is described in further detail in concurrently filed U.S. patent application no. 09/797,198, entitled "Systems and Methods for Management of Memory" by Qiu et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 23 from lines 27 - 31 and on page 24 from lines 1 - 5 with the following:

--For increasing delivery efficiency of continuous content, such as streaming multimedia content, storage management engine 1040 may employ caching algorithms that consider the dynamic characteristics of continuous content. Suitable examples include, but are not limited to, interval caching algorithms. In one embodiment, improved caching performance of continuous content may be achieved using an LMLRU caching algorithm that weighs ongoing viewer cache value versus the dynamic time-size cost of maintaining particular content in cache memory. Such a caching algorithm is described in further detail in concurrently filed U.S. patent application no. 09/797,201, entitled "Systems and Methods for Management of Memory in Information Delivery Environments" by Qiu et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 46 from lines 10 - 16 with the following:

--Information gathered by monitoring agents 245 may be employed for a wide variety of purposes including for billing of individual content suppliers and/or users for pro-rata use of one or more resources, resource use analysis and optimization, resource health alarms, *etc.* In addition, monitoring agents may be employed to enable the deterministic delivery of content by

system 200 as described in concurrently filed, co-pending United States patent application number 09/797,200, entitled "Systems and Methods for the Deterministic Management of Information," which is incorporated herein by reference.--

Please replace the paragraph on page 28 from lines 16 - 26 with the following:

--In yet other embodiments, the processing modules may be specialized to specific applications, for example, for processing and delivering HTTP content, processing and delivering RTSP content, or other applications. For example, in such an embodiment an application processing module 1070a and storage processing module 1040a may be specially programmed for processing a first type of request received from a network. In the same system, application processing module 1070b and storage processing module 1040b may be specially programmed to handle a second type of request different from the first type. Routing of requests to the appropriate respective application and/or storage modules may be accomplished using a distributive interconnect and may be controlled by transport and/or interface processing modules as requests are received and processed by these modules using policies set by the system management engine.--

Please replace the paragraph on page 34 from lines 2 - 13 with the following:

--In yet another embodiment, at least two network interface modules 1030a and 1030b may be provided, as illustrated in FIG. 1A. In this embodiment, a first network interface engine 1030a may receive incoming data from a network and pass the data directly to the second network interface engine 1030b for transport back out to the same or different network. For example, in the remote or live broadcast application described above, first network interface engine 1030a may receive content, and second network interface engine 1030b provide the content to the network 1020 to fulfill requests from one or more clients for this content. Peer-to-peer level communication between the two network interface engines allows first network interface engine 1030a to send the content directly to second network interface engine 1030b via distributive interconnect 1080. If necessary, the content may also be routed through transport

processing engine 1050, or through transport processing engine 1050 and application processing engine 1070, in a manner described above.--

Please replace the paragraph on page 34 from lines 23 - 31 with the following:

--The content delivery system 1010 described above is configured in a peer-to-peer manner that allows the various engines and modules to communicate with each other directly as peers through the distributed interconnect. This is contrasted with a traditional server architecture in which there is a main CPU. Furthermore unlike the arbitrated bus of traditional servers, the distributed interconnect 1080 provides a switching means which is not arbitrated and allows multiple simultaneous communications between the various peers. The data and communication flow may by-pass unnecessary peers such as the return of data from the storage management processing engine 1040 directly to the network interface processing engine 1030 as described with reference to FIG. 1B.--

Please replace the paragraph on page 42 from lines 23 - 31 with the following:

--It will be understood with benefit of this disclosure that the hardware embodiment and multiple engine configurations thereof illustrated in FIGS. 1C-1F are exemplary only, and that other hardware embodiments and engine configurations thereof are also possible. It will further be understood that in addition to changing the assignments of individual processing modules to particular processing engines, distributive interconnect 1080 enables the various processing flow paths between individual modules employed in a particular engine configuration in a manner as described in relation to FIG. 1B. Thus, for any given hardware embodiment and processing engine configuration, a number of different processing flow paths may be employed so as to optimize system performance to suit the needs of particular system applications.--

Please replace the paragraph on page 44 from lines 15 - 28 with the following:

FIG. 2 illustrates a single networking subsystem processor module 205 that is provided to perform the combined functions of network interface processing engine 1030 and transport processing engine 1050 of FIG. 1A. Communication and content delivery between network 260 and networking subsystem processor module 205 are made through network connection 270. For certain applications, the functions of network interface processing engine 1030 and transport processing engine 1050 of FIG. 1A may be so combined into a single module 205 of FIG. 2 in order to reduce the level of communication and data traffic handled by communications path 230 and data movement path 235 (or single switch fabric), without adversely impacting the resources of application processing engine or subsystem module. If such a modification were made to the system of FIG. 1A, content requests may be passed directly from the combined interface/transport engine to network application processing engine 1070 via distributive interconnect 1080. Thus, as previously described the functions of two or more separate content delivery system engines may be combined as desired (e.g., in a single module or in multiple modules of a single processing blade), for example, to achieve advantages in efficiency or cost.--

In the claims:

Please cancel claims 1-132 and add the following new claims:

--133. A data storage system, comprising:

- at least one system processor comprising a storage processor;
- a system interface connection configured to be coupled to a network;
- at least one network processor, the network processor coupled to the system interface connection to receive data from the network; and
- an interconnection between the system processor and the network processor so that the network processor may analyze data provided from the network and process the data at least in part and then forward the data to the interconnection so that other processing may be performed on the data within the system so that storage system functionality may be accomplished.

134. The data storage system of claim 133, further comprising an application processor.
135. The data storage system of claim 133, wherein the system comprises a plurality of system processors configured as an asymmetric multi-processor system.
136. The data storage system of claim 133, wherein the system comprises a plurality of processors communicating in a peer to peer environment.
137. The data storage system of claim 136, wherein the plurality of processors comprises the network processor and the system processor.
138. The data storage system of claim 137, wherein the plurality of processors comprises the network processor and multiple system processors.
139. The data storage system of claim 138, wherein the multiple system processors comprises a storage processor and an application processor.
140. The data storage system of claim 139, wherein the interconnection comprises a distributed interconnection.
141. data storage system of claim 140, wherein the distributed interconnection comprises a switch fabric.
142. The data storage system of claim 136, wherein the interconnection comprises a distributed interconnection.
143. The data storage system of claim 142, wherein the distributed interconnection comprises a switch fabric.

144. The data storage system of claim 143, further comprising a storage device coupled to the storage processor.

145. The data storage system of claim 133, wherein the interconnection comprises a switch fabric.

146. The data storage system of claim 145, further comprising a storage device coupled to the storage processor.

147. The data storage system of claim 133, wherein the network processor filters data incoming to the data storage system from the network.

148. The data storage system of claim 133, the network processor enabling accelerated system performance.

149. The data storage system of claim 133, the data storage system being a content delivery system.

150. The data storage system of claim 149, the data storage system providing accelerated content delivery.

151. The data storage system of claim 133, further comprising a storage device coupled to the storage processor.

152. A method of operating a data storage system, the method comprising:
providing a network processor within the data storage system, the network processor
being configured to be coupled to an interface which couples the data storage
system to a network;
processing data passing through the interface with the network processor; and

forwarding data from the network processor to a system processor which then performs at least some data storage system functionality upon the data.

153. The method of claim 152, wherein the network processor analyzes headers of data packets transmitted to the data storage system from the network.

154. The method of claim 153, the method further comprising configuring the network processor and the system processor in a peer to peer computing environment.

155. The method of claim 153, wherein the data storage system comprises a plurality of system processors, the method further comprising configuring the network processor and the plurality of system processors in a peer to peer computing environment.

156. The method of claim 155, the network processor and the plurality of system processors configured as an asymmetric multi-processor manner.

157. The method of claim 156, the method further comprising operating the data storage system in a staged pipeline processing manner.

158. The method of claim 157, the plurality of system processors comprising a storage processor and an application processor.

159. The method of claim 157, wherein the data storage system functionality is content delivery.

160. The method of claim 159, the method further comprising accelerating the content delivery of the data storage system.

161. The method of claim 152, the method further comprising configuring the network processor and the system processor in a peer to peer computing environment.

162. The method of claim 152, wherein the data storage system comprises a plurality of system processors, the method further comprising configuring the network processor and the plurality of system processors in a peer to peer computing environment.

163. The method of claim 152, the data storage system configured as an asymmetric multi-processor system.

164. The method of claim 152, the network processor performing filter functions upon the data passing through the interface.

165. The method of claim 152, the data forwarded by the network processor being forwarded through a switch fabric.

166. The method of claim 157, wherein the data storage system functionality is content delivery, the method further comprising accelerating the content delivery of the network endpoint system.

167. A method of providing a data storage system through the use of a network connectable computing system, comprising:

- providing a plurality of separate processor engines, the processor engines being assigned separate tasks in an asymmetrical multi-processor configuration;
- providing a storage processor engine, the storage processor engine being one of the plurality of separate processor engines;
- providing a network interface connection to at least one of the processor engines to couple the data storage system to a network;
- providing a storage interface connection to the storage processor engine to couple the storage processor engine to storage device; and
- accelerating content delivery through the data storage system.

168. The method of claim 167, wherein the separate processor engines and the storage processor engine communicate as peers in a peer to peer environment.
169. The method of claim 168, wherein the separate processors and the storage processor engine communicate across a distributed interconnect.
170. The method of claim 169, wherein the distributed interconnect is a switch fabric.
171. The method of claim 169, wherein the processor engine coupling the data storage system to a network comprises a network processor.
172. The method of claim 171, further comprising performing look ahead processing within the network processor to off-load processing tasks from the other processor engines.
173. The method of claim 167, wherein the separate processor engine coupling the data storage system to a network is a network interface processor engine comprising a network processor.
174. The method of claim 173, further comprising performing look ahead processing within the network processor to off-load processing tasks from the other processor engines.
175. The method of claim 174, wherein the separate processor engines and the storage processor engine communicate as peers in a peer to peer environment. \
176. The method of claim 175, wherein the separate processor engines and the storage processor engine communicate across a distributed interconnect.
177. The method of claim 176, wherein the distributed interconnect is a switch fabric.

178. The method of claim 175, wherein one of the separate processor engines is an application processor engine.
179. The method of claim 178, wherein the network interface processor engine, the storage processor engine and the application processor engine communicate as peers in a peer to peer environment.
180. The method of claim 179, further comprising performing at least some system management functions in a system management processor engine.
181. The method of claim 180, further comprising tracking system performance within the system management processor engine.
182. The method of claim 181, further comprising implementing system policies with the system management processor engine.
183. A network connectable data storage system, comprising:
a first processor engine;
a second processor engine, the second processor engine being assigned types of tasks different from the types of tasks assigned to the first processor engine;
a storage processor engine, the storage processor engine being assigned types of tasks that are different from the types of tasks assigned to the first and second processor engines, the storage processor engine being configured to be coupled to a data storage device; and
a distributed interconnection coupled to the first, second and third processor engines, the tasks of the first, second and third processor engines being assigned such that the system operates in staged pipeline manner through the distributed interconnection.
184. The system of claim 183, wherein the system performs at least some network endpoint functionality.

195. The system of claim 193, wherein the first processor engine is a network interface processor engine comprising a network processor.
196. The system of claim 195, wherein the network interface processor engine, the application processor engine, and the storage processor engine communicate in a peer to peer fashion.
197. A network connectable data storage system, comprising:
- a first processor engine;
 - a second processor engine, the second processor engine being assigned types of tasks different from the types of tasks assigned to the first processor engine;
 - a storage processor engine, the storage processor engine being assigned types of tasks that are different from the types of tasks assigned to the first and second processor engines;
 - a data storage device, the storage processor engine being configured to be coupled to the data storage device; and
 - a distributed interconnection coupled to the first, second and third processor engines, the tasks of the first, second and third processor engines being assigned such that the system operates in staged pipeline manner through the distributed interconnection, wherein at least one of the first or second processor engines performs system management functions so as to off-load management functions from the other processor engines.
198. The system of claim 197, wherein the first processor engine is a system management processor engine that performs at least some of the off-loaded management functions.
199. The system of claim 197, wherein the first processor engine is a network interface processor engine that performs at least some of the off-loaded management functions.

20150404T12360

- 200. The system of claim 199, wherein the network interface processor engine comprises a network processor.
- 201. The system of claim 197, wherein at least some system management functions are off-loaded from the storage processor engine.
- 202. The system of claim 201, wherein the second processor engine is an application processor engine, wherein at least some system management functions are off-loaded from both the storage processor engine and the application processor engine.
- 203. The system of claim 202, wherein the system management functions comprise prioritizing data flow through the system.
- 204. The system of claim 202, wherein the system management functions comprise quality of service functions.
- 205. The system of claim 202, wherein the system management functions comprise service level agreement functions.
- 206. The system of claim 202, wherein the system management functions comprise filtering content requests.
- 207. The system of claim 206, wherein the first processor engine is a system management processor engine that performs the filtering functions.
- 208. The system of claim 206, wherein the first processor engine is a network interface processor engine that performs the filtering functions, the network interface processor engine comprising a network processor.

209. The system of claim 197, wherein the system management functions comprise prioritizing data flow through the system.

210. The system of claim 197, wherein the system management functions comprise quality of service functions.

211. The system of claim 197, wherein the system management functions comprise service level agreement functions.

212. The system of claim 197, wherein the system management functions comprise filtering content requests.

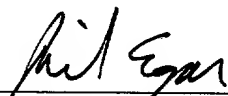
213. The system of claim 212, wherein the first processor engine is a system management processor engine that performs the filtering functions.

214. The system of claim 212, wherein the first processor engine is a network interface processor engine that performs the filtering functions, the network interface processor engine comprising a network processor.--

REMARKS

Should any fees under 37 CFR 1.16-1.21 be required for any reason relating to the enclosed materials, the Commissioner is authorized to deduct such fees from Deposit Account No. 10-1205/SURG:157. The examiner is invited to contact the undersigned at the phone number indicated below with any questions or comments, or to otherwise facilitate expeditious and compact prosecution of the application.

Respectfully submitted,



Richard D. Egan
Registration No. 36,788
Attorney for Applicant

O'KEEFE, EGAN & PETERMAN
1101 Capital of Texas Highway South
Building C, Suite 200
Austin, Texas 78746
(512) 347-1611
FAX: (512) 347-1615

APPENDIX
MARKED UP VERSION OF AMENDMENTS
AS REQUIRED BY RULE 121

In The Specification:

On page 1, please replace the paragraph from lines 1-2 with:

--Patent Application for
**“NETWORK CONNECTED COMPUTING SYSTEM INCLUDING STORAGE
SYSTEM”** --

On page 1, please replace the paragraph from lines 6-22 with:

This application is a continuation of Application Serial No. 09/797,413, which was filed March 1, 2001 and is entitled “NETWORK CONNECTED COMPUTING SYSTEM”, which in turn [This application] claims priority from Provisional Application Serial No. 60/187,211, which was filed March 3[0], 2000 and is entitled “SYSTEM AND APPARATUS FOR INCREASING FILE SERVER BANDWIDTH,” from Provisional Application Serial No. 60/246,343, which was filed November 7, 2000 and is entitled “NETWORK CONTENT DELIVERY SYSTEM WITH PEER TO PEER PROCESSING COMPONENTS,” from Provisional Application Serial No. 60/246,335, which was filed November 7, 2000 and is entitled “NETWORK SECURITY ACCELERATOR,” from Provisional Application Serial No. 60/246,443, which was filed November 7, 2000 and is entitled “METHODS AND SYSTEMS FOR THE ORDER SERIALIZATION OF INFORMATION IN A NETWORK PROCESSING ENVIRONMENT,” from Provisional Application Serial No. 60/246,373, which was filed November 7, 2000 and is entitled “INTERPROCESS COMMUNICATIONS WITHIN A NETWORK NODE USING SWITCH FABRIC,” from Provisional Application Serial No. 60/246,444, which was filed November 7, 2000 and is entitled “NETWORK TRANSPORT ACCELERATOR,” and from Provisional Application Serial No. 60/246,372, which was filed November 7, 2000 and is entitled “SINGLE CHASSIS NETWORK ENDPOINT SYSTEM WITH NETWORK PROCESSOR FOR LOAD BALANCING,” the disclosures of each being incorporated herein by reference.

Please replace the paragraph on page 16 from lines 10 -31 and on page 17 from lines 1-3 with the following:

--The processing cores of a network processor are typically accompanied by special purpose cores that perform specific tasks, such as fabric interfacing, table lookup, queue management, and buffer management. Network processors typically have their memory management optimized for data movement, and have multiple I/O and memory buses. The programming capability of network processors permit them to be programmed for a variety of tasks, such as load balancing, network protocol processing, network security policies, and QoS/CoS support. These tasks can be tasks that would otherwise be performed by another processor. For example, TCP/IP processing may be performed by a network processor at the front end of an endpoint system. Another type of processing that could be offloaded is execution of network security policies or protocols. A network processor could also be used for load balancing. Network processors used in this manner can be referred to as "network accelerators" because their front end "look ahead" processing can vastly increase network response speeds. Network processors perform look ahead processing by operating at the front end of the network endpoint to process network packets in order to reduce the workload placed upon the remaining endpoint resources. Various uses of network accelerators are described in the following concurrently filed U.S. patent applications: no. []09/797,412, entitled "Network Transport Accelerator," by Bailey et. al; no. []09/797,507, entitled "Single Chassis Network Endpoint System With Network Processor For Load Balancing," by Richter et. al; and no. []09/797,411, entitled "Network Security Accelerator," by Canion et. al; the disclosures of which are all incorporated herein by reference. When utilizing network processors in an endpoint environment it may be advantageous to utilize techniques for order serialization of information, such as for example, as disclosed in concurrently filed U.S. patent application no. []09/797,197, entitled "Methods and Systems For The Order Serialization Of Information In A Network Processing Environment," by Richter et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 23 from lines 12 - 25 with the following:

--Storage management engine 1040 may employ any suitable method for caching data, including simple computational caching algorithms such as random removal (RR), first-in first-out (FIFO), predictive read-ahead, over buffering, etc. algorithms. Other suitable caching algorithms include those that consider one or more factors in the manipulation of content stored within the cache memory, or which employ multi-level ordering, key based ordering or function based calculation for replacement. In one embodiment, storage management engine may implement a layered multiple LRU (LMLRU) algorithm that uses an integrated block/buffer management structure including at least two layers of a configurable number of multiple LRU queues and a two-dimensional positioning algorithm for data blocks in the memory to reflect the relative priorities of a data block in the memory in terms of both recency and frequency. Such a caching algorithm is described in further detail in concurrently filed U.S. patent application no. []09/797,198, entitled "Systems and Methods for Management of Memory" by Qiu et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 23 from lines 27 - 31 and on page 24 from lines 1 - 5 with the following:

--For increasing delivery efficiency of continuous content, such as streaming multimedia content, storage management engine 1040 may employ caching algorithms that consider the dynamic characteristics of continuous content. Suitable examples include, but are not limited to, interval caching algorithms. In one embodiment, improved caching performance of continuous content may be achieved using an LMLRU caching algorithm that weighs ongoing viewer cache value versus the dynamic time-size cost of maintaining particular content in cache memory. Such a caching algorithm is described in further detail in concurrently filed U.S. patent application no. []09/797,201, entitled "Systems and Methods for Management of Memory in Information Delivery Environments" by Qiu et. al, the disclosure of which is incorporated herein by reference.--

Please replace the paragraph on page 46 from lines 10 - 16 with the following:

--Information gathered by monitoring agents 245 may be employed for a wide variety of purposes including for billing of individual content suppliers and/or users for pro-rata use of one or more resources, resource use analysis and optimization, resource health alarms, *etc.* In addition, monitoring agents may be employed to enable the deterministic delivery of content by system 200 as described in concurrently filed, co-pending United States patent application number []09/797,200, entitled "Systems and Methods for the Deterministic [Delivery of Data and Services]Management of Information," which is incorporated herein by reference.--

Please replace the paragraph on page 28 from lines 16 - 26 with the following:

--In yet other embodiments, the processing modules may be specialized to specific applications, for example, for processing and delivering HTTP content, processing and delivering RTSP content, or other applications. For example, in such an embodiment an application processing module 1070a and storage processing module 10[5]40a may be specially programmed for processing a first type of request received from a network. In the same system, application processing module 1070b and storage processing module 10[5]40b may be specially programmed to handle a second type of request different from the first type. Routing of requests to the appropriate respective application and/or storage modules may be accomplished using a distributive interconnect and may be controlled by transport and/or interface processing modules as requests are received and processed by these modules using policies set by the system management engine.--

Please replace the paragraph on page 34 from lines 2 - 13 with the following:

--In yet another embodiment, at least two network interface modules 1030a and 1030b may be provided, as illustrated in FIG. 1A. In this embodiment, a first network interface engine 1030a may receive incoming data from a network and pass the data directly to the second

network interface engine 1030b for transport back out to the same or different network. For example, in the remote or live broadcast application described above, first network interface engine 1030a may receive content, and second network interface engine 1030b provide the content to the network 1020 to fulfill requests from one or more clients for this content. Peer-to-peer level communication between the two network interface engines allows first network interface engine 1030a to send the content directly to second network interface engine 1030b via distributive interconnect 1080. If necessary, the content may also be routed through transport processing engine 1050, or through [network]transport processing engine 1050 and application processing engine 1070, in a manner described above.--

Please replace the paragraph on page 34 from lines 23 - 31 with the following:

--The content delivery system 1010 described above is configured in a peer-to-peer manner that allows the various engines and modules to communicate with each other directly as peers through the distributed interconnect. This is contrasted with a traditional server architecture in which there is a main CPU. Furthermore unlike the arbitrated bus of traditional servers, the distributed interconnect 1080 provides a switching means which is not arbitrated and allows multiple simultaneous communications between the various peers. The data and communication flow may by-pass unnecessary peers such as the return of data from the storage management processing engine 10[6]40 directly to the network interface processing engine 1030 as described with reference to FIG. 1B.--

Please replace the paragraph on page 42 from lines 23 - 31 with the following:

--It will be understood with benefit of this disclosure that the hardware embodiment and multiple engine configurations thereof illustrated in FIGS. 1C-1F are exemplary only, and that other hardware embodiments and engine configurations thereof are also possible. It will further be understood that in addition to changing the assignments of individual processing modules to particular processing engines, distributive interconnect 1080 enables the [vary]various processing flow paths between individual modules employed in a particular engine configuration in a

manner as described in relation to FIG. 1B. Thus, for any given hardware embodiment and processing engine configuration, a number of different processing flow paths may be employed so as to optimize system performance to suit the needs of particular system applications.--

Please replace the paragraph on page 44 from lines 15 - 28 with the following:

--FIG. 2 illustrates a single networking subsystem processor module 205 that is provided to perform the combined functions of network interface processing engine 1030 and transport processing engine 1040 of FIG. 1A. Communication and content delivery between network 260 and networking subsystem processor module 205 are made through network connection 270. For certain applications, the functions of network interface processing engine 1030 and transport processing engine 1050 of FIG. 1A may be so combined into a single module 205 of FIG. 2 in order to reduce the level of communication and data traffic handled by communications path 230 and data movement path 235 (or single switch fabric), without adversely impacting the resources of application processing engine or subsystem module. If such a modification were made to the system of FIG. 1A, content requests may be passed directly from the combined interface/transport engine to network application processing engine 1070 via distributive interconnect 1080. Thus, as previously described the functions of two or more separate content delivery system engines may be combined as desired (*e.g.*, in a single module or in multiple modules of a single processing blade), for example, to achieve advantages in efficiency or cost.--

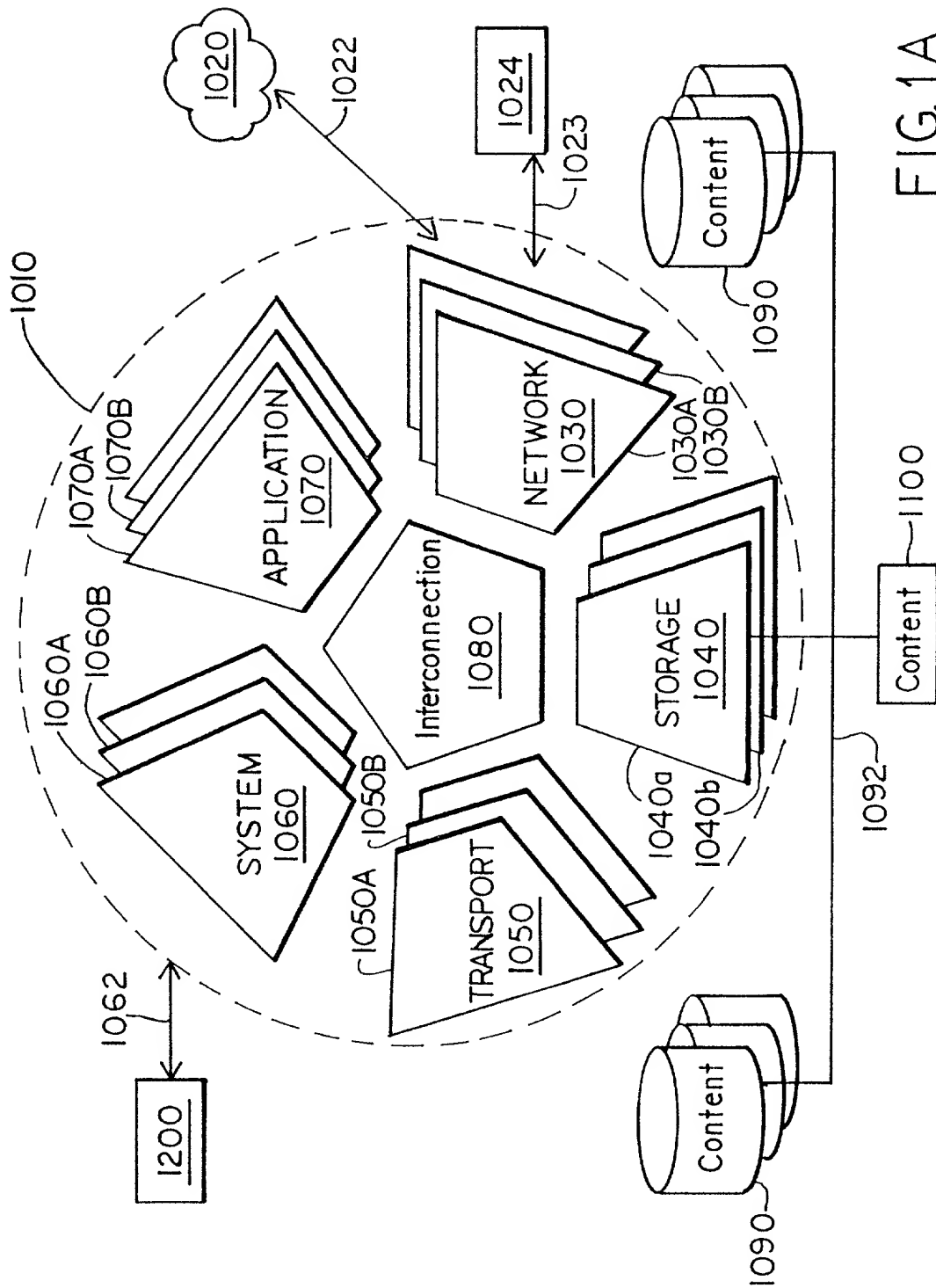


FIG. 1A



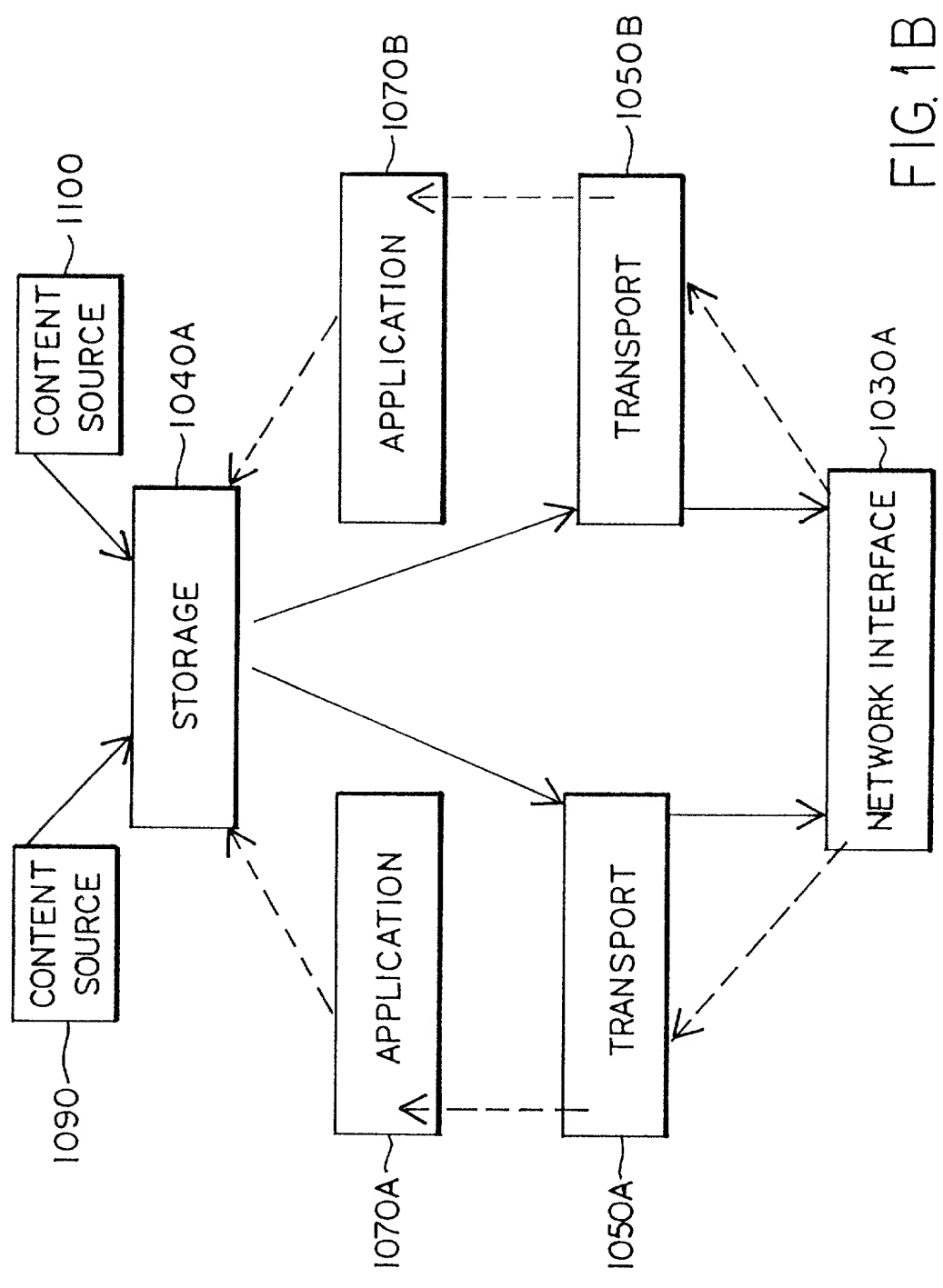


FIG. 1B

FIG. 1C^I FIG. 1C^{II}

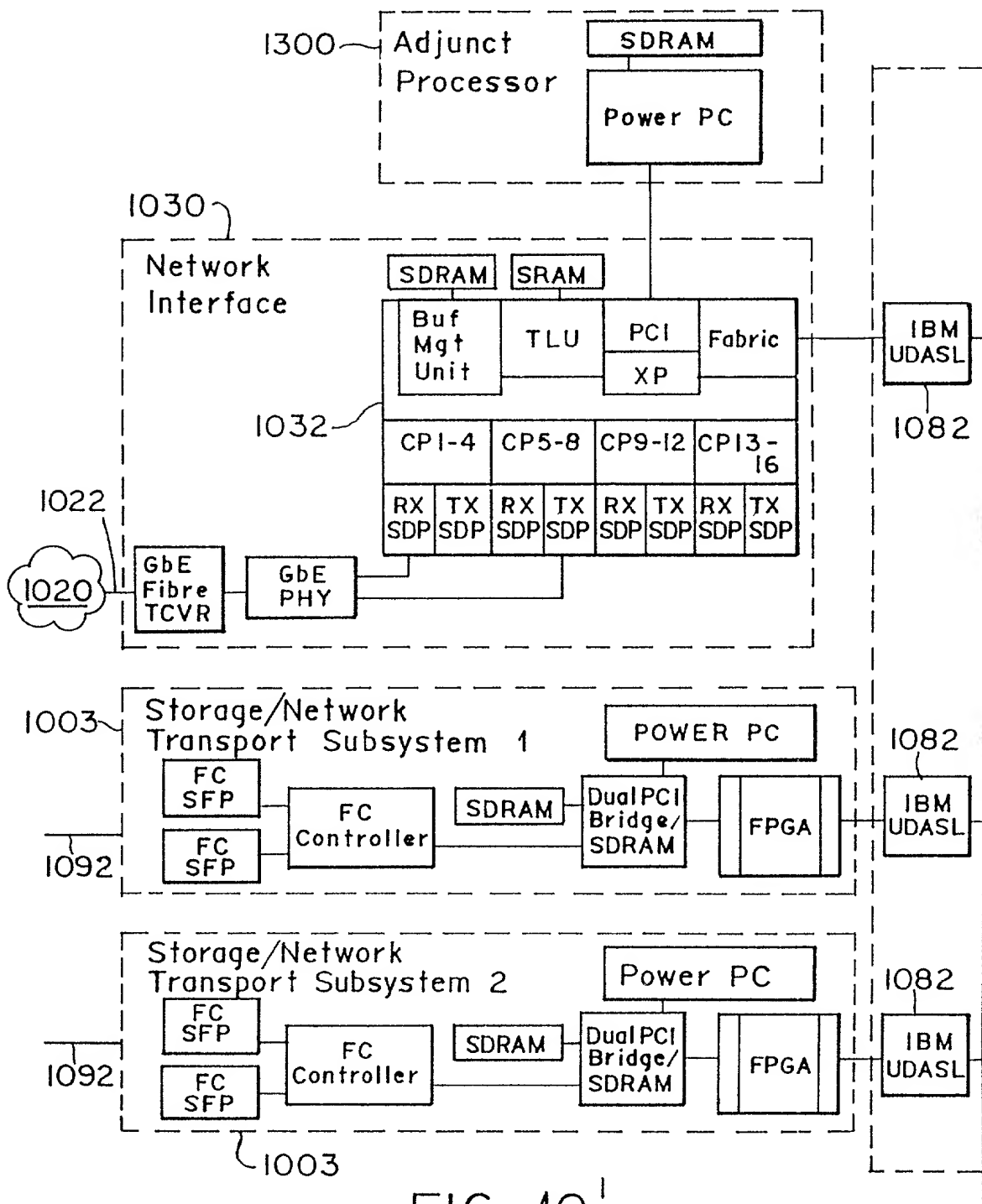


FIG. 1C^I

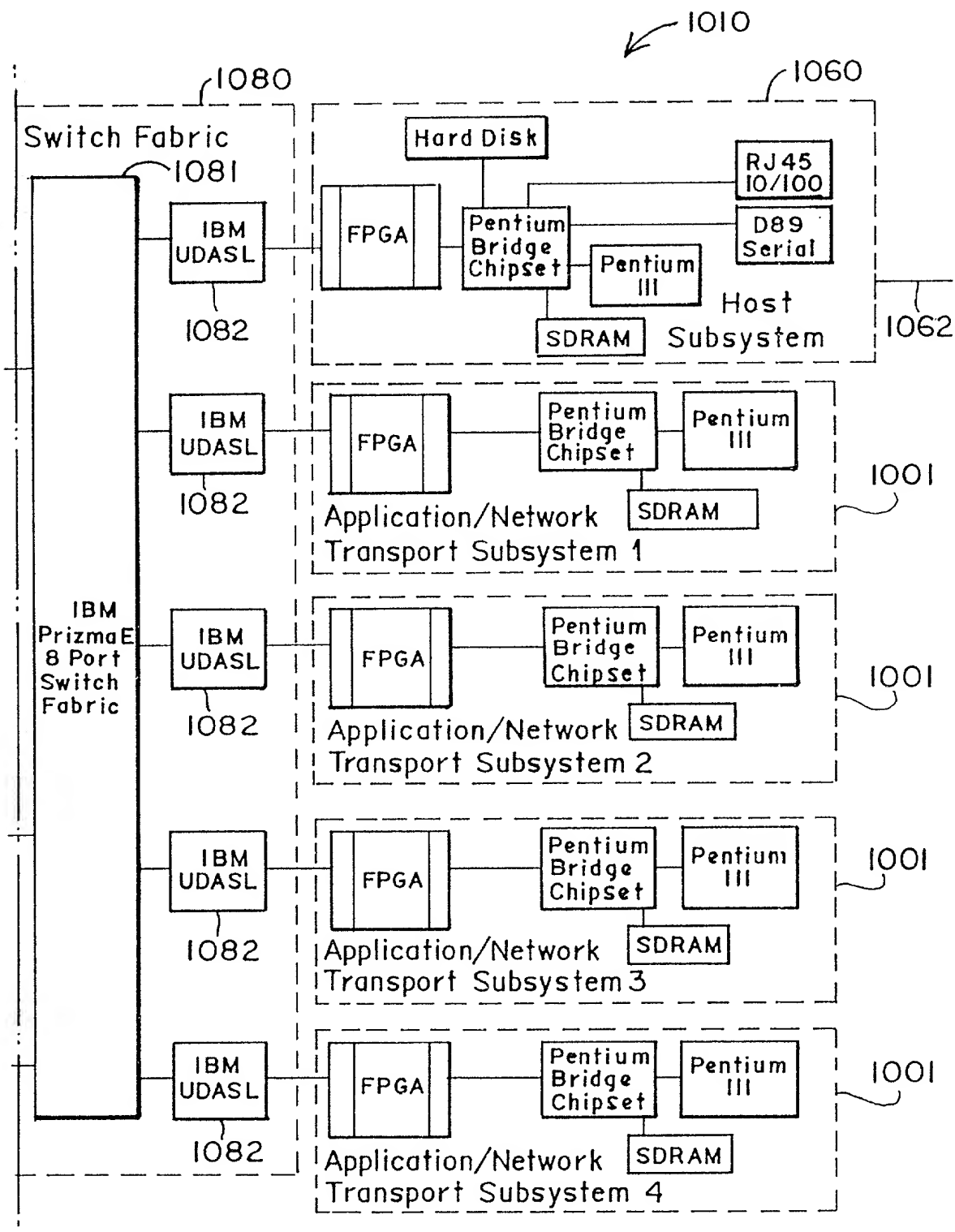
FIG. 1C¹¹

FIG. 1D FIG. 1D^I FIG. 1D^{II}

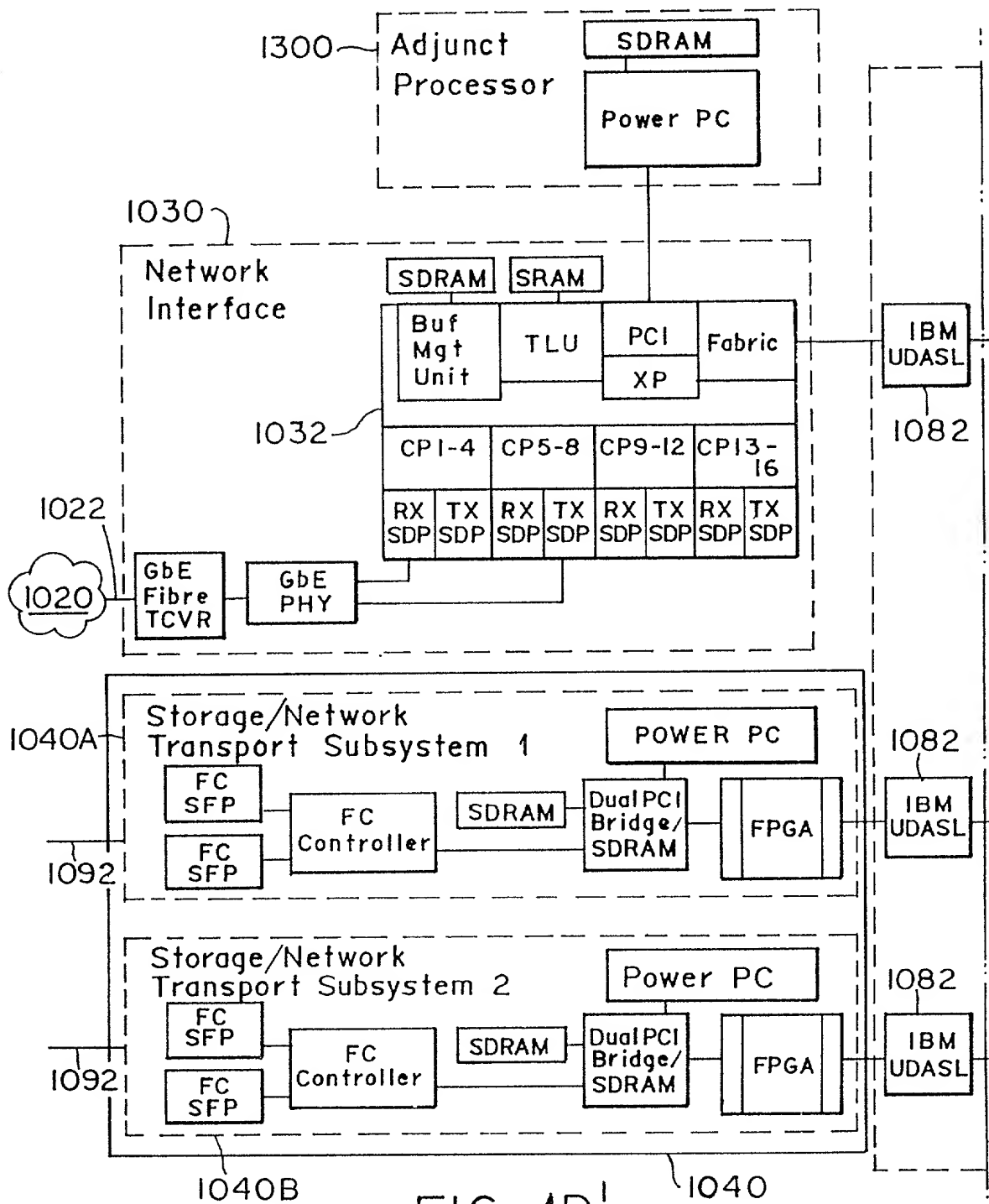


FIG. 1D^I

0987434.053404
TOP SECRET 2860

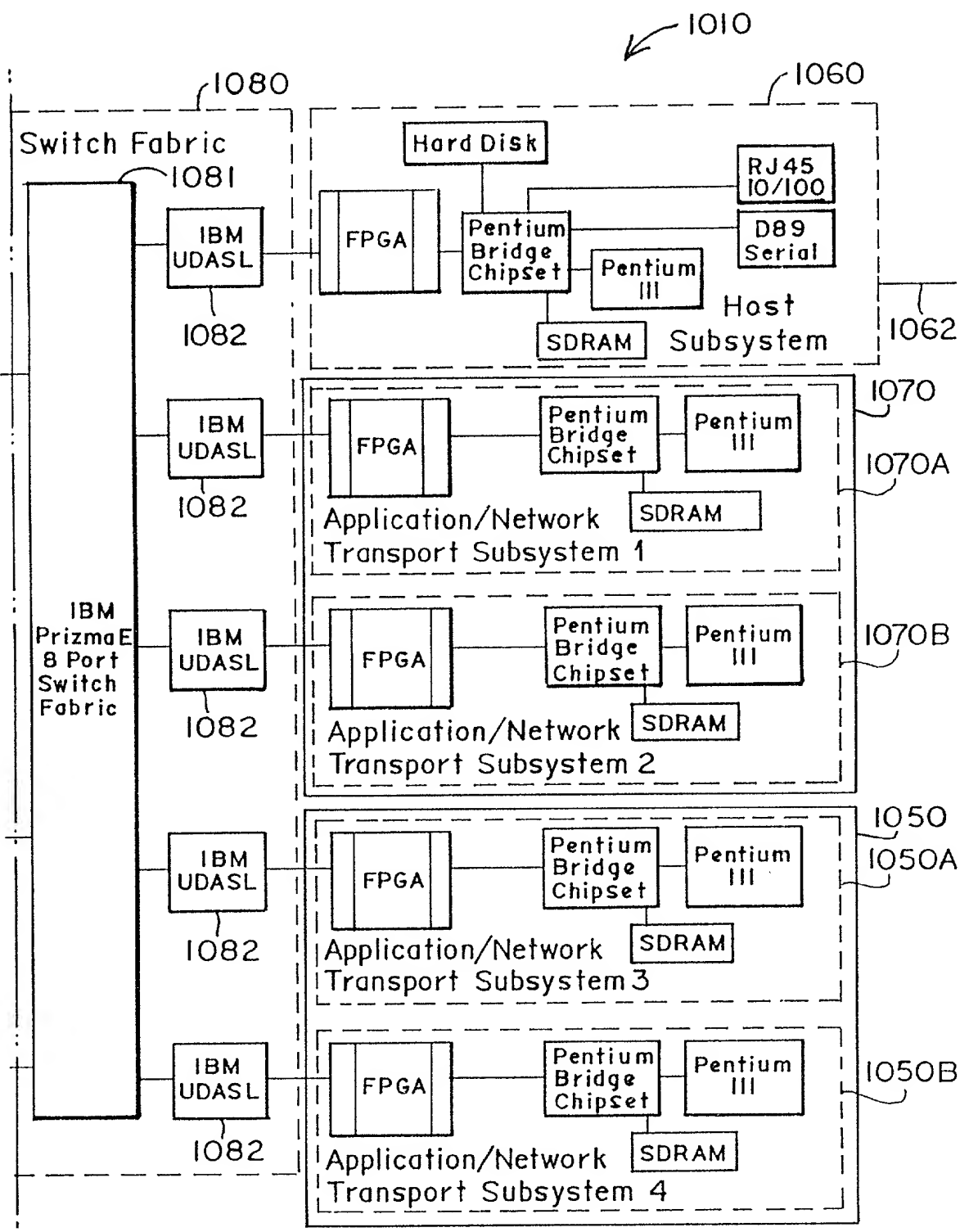


FIG. 1D''

FIG. 1E FIG. 1E' FIG. 1E''

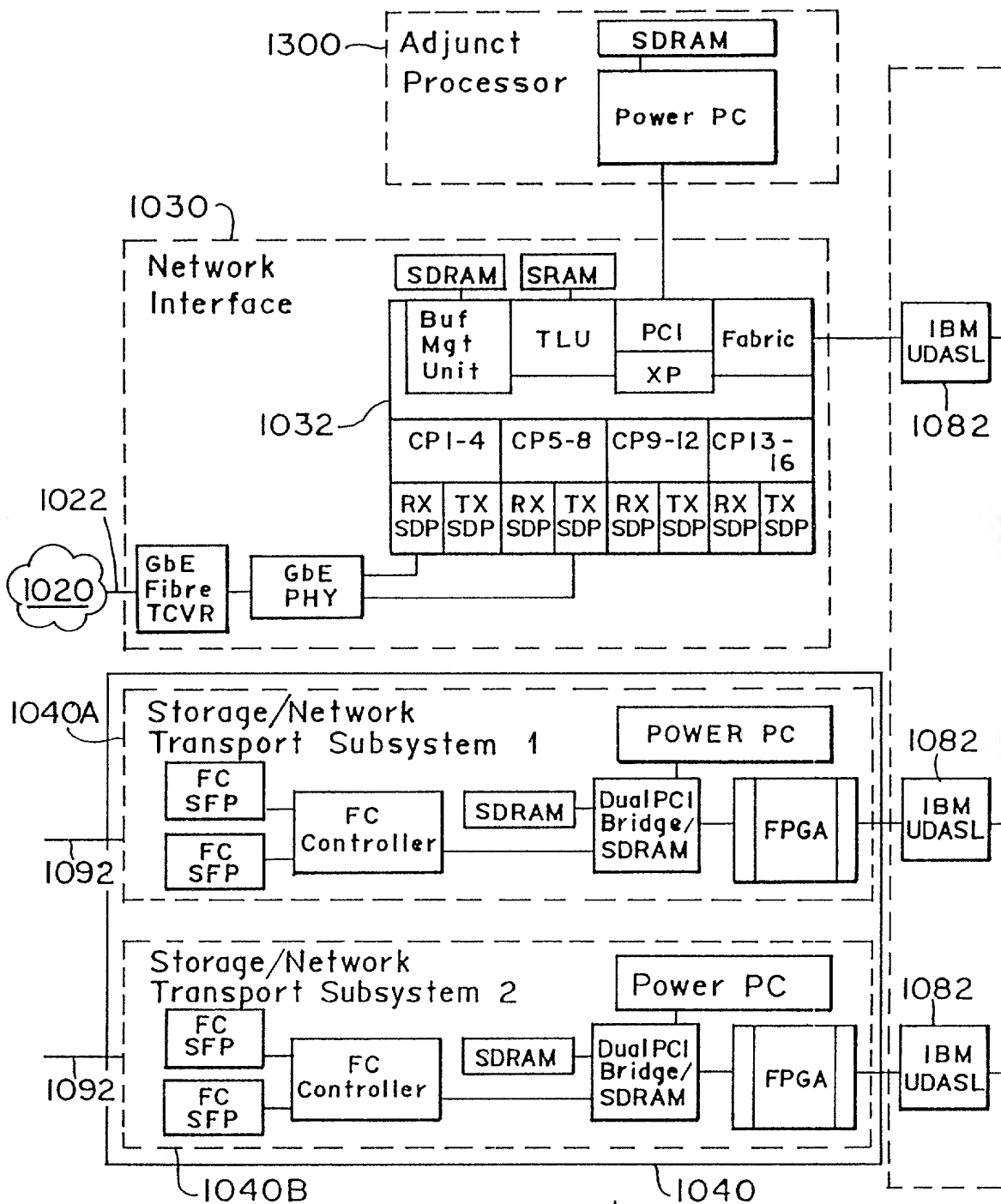


FIG. 1E'

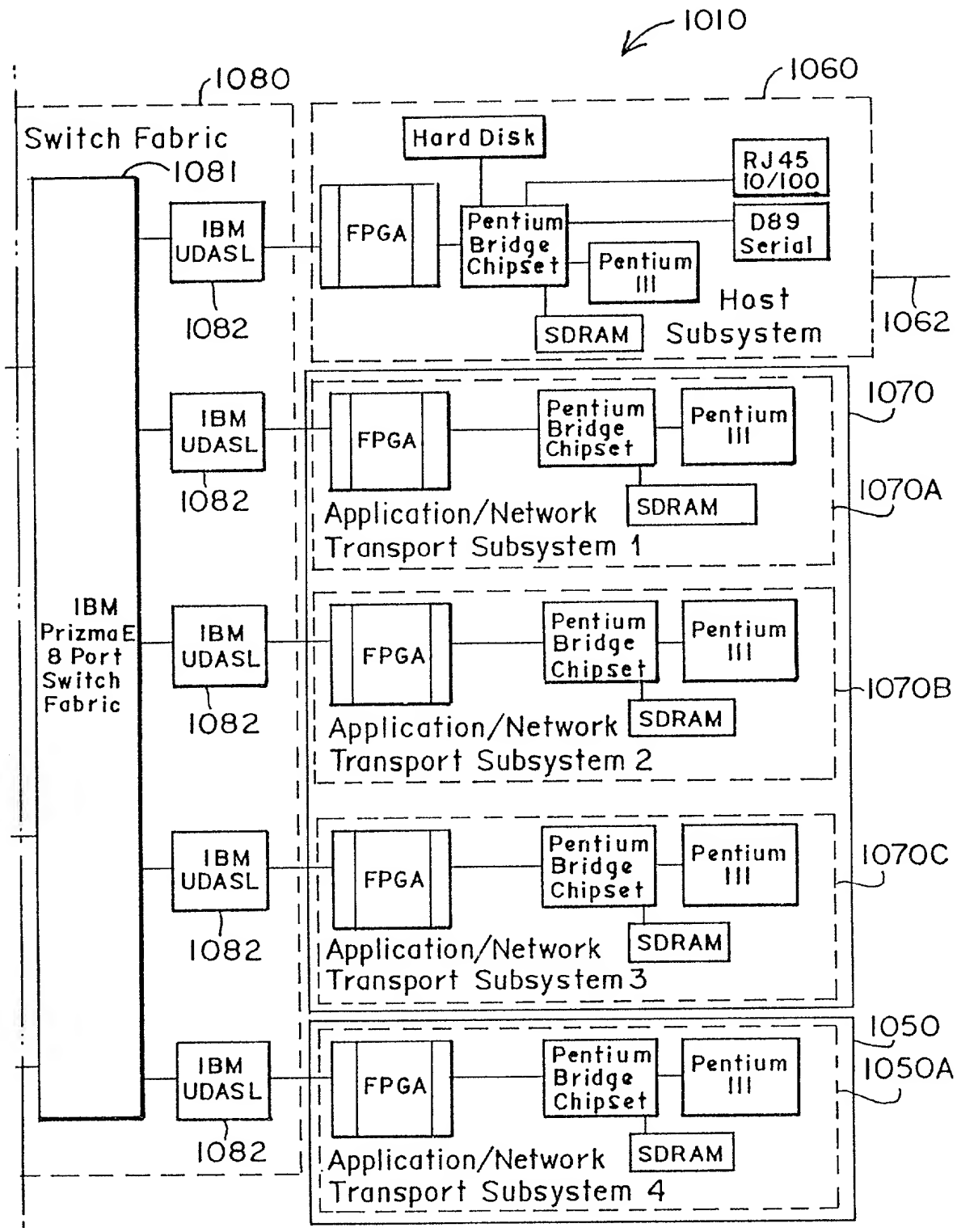


FIG. 1E^{II}

FIG. 1F

FIG. 1F^I

FIG. 1F^{II}

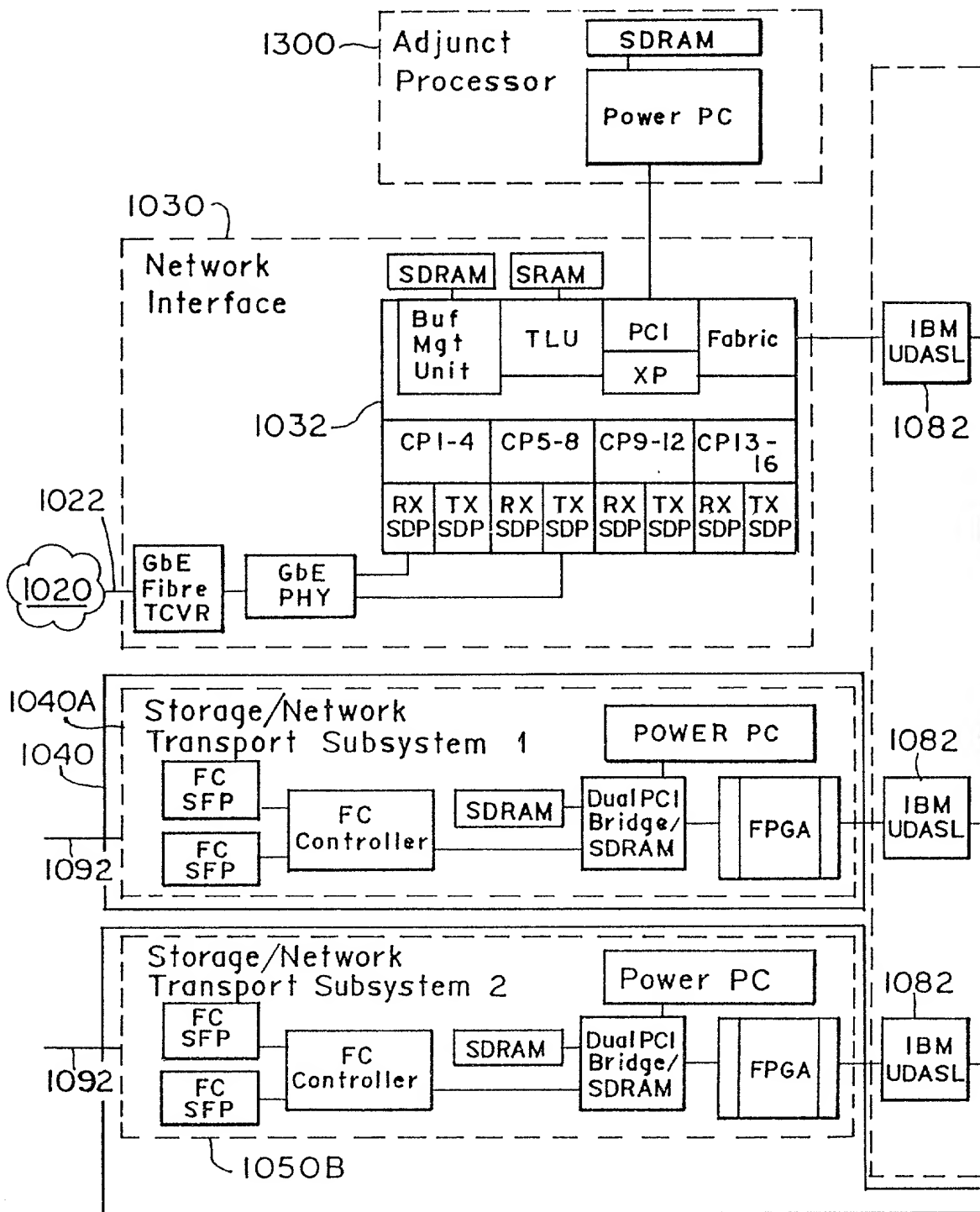


FIG. 1F^I

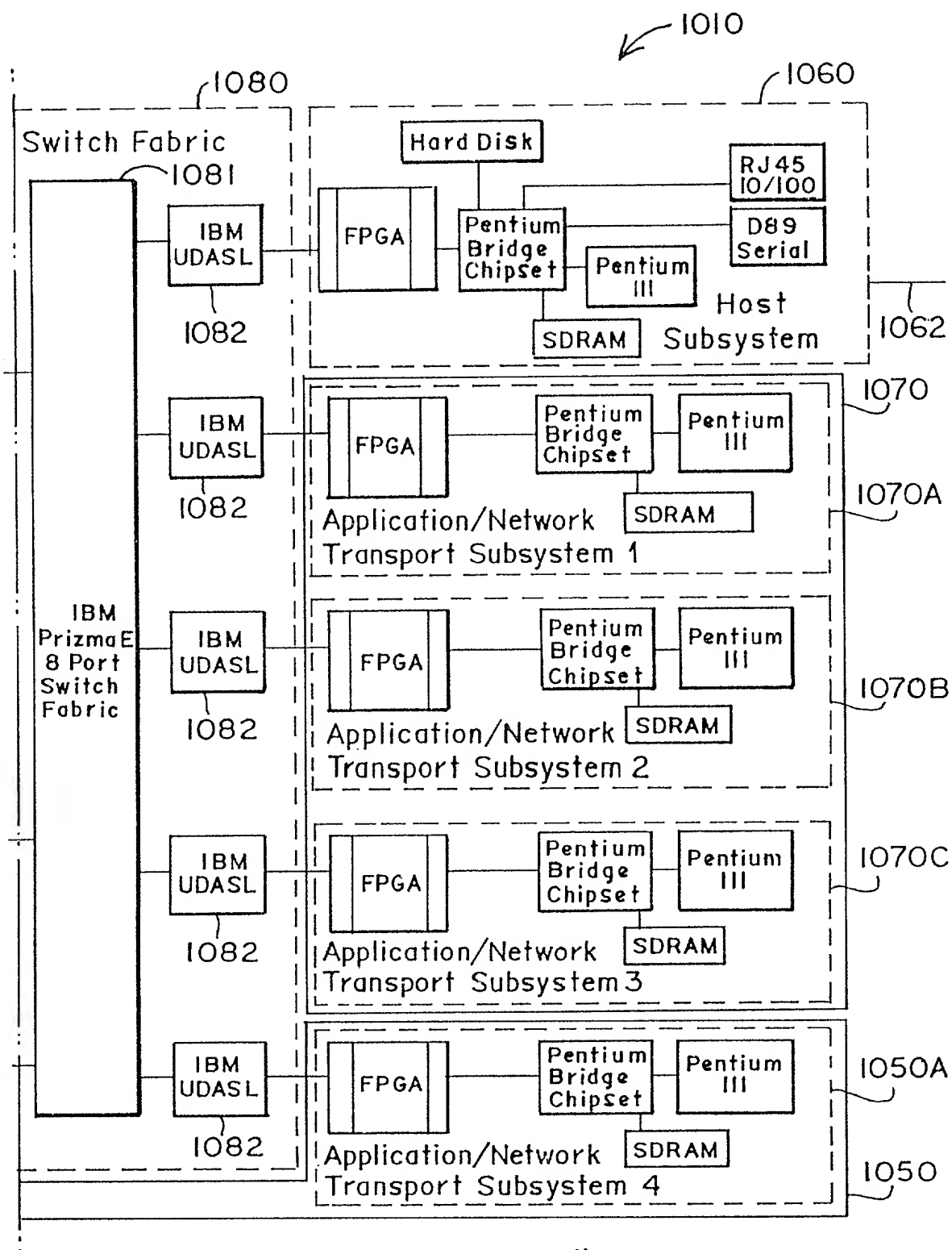


FIG. 1F^{II}

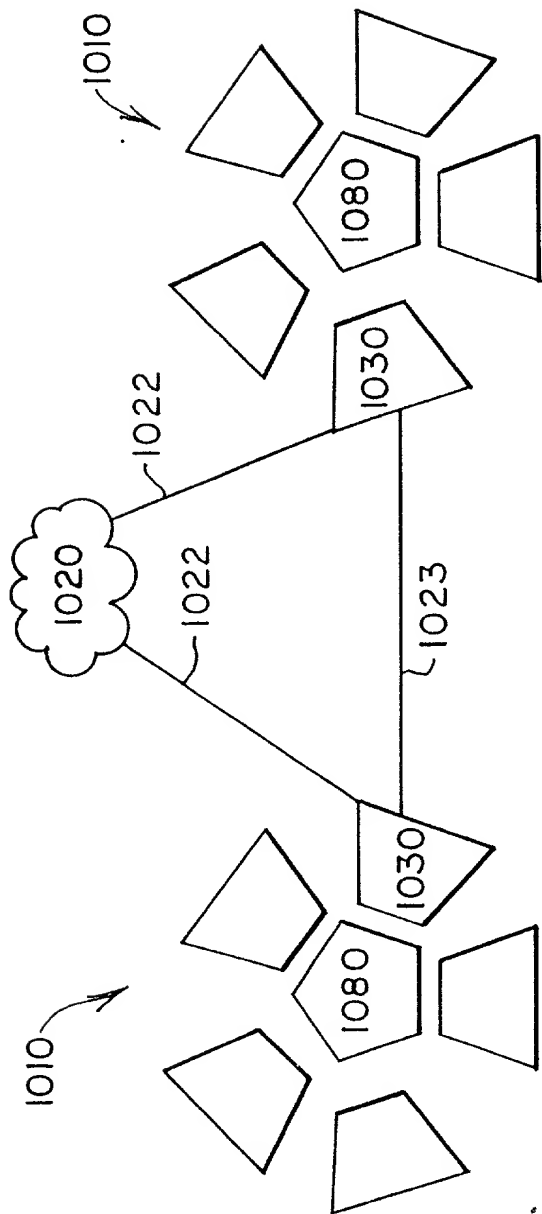


FIG. 1G

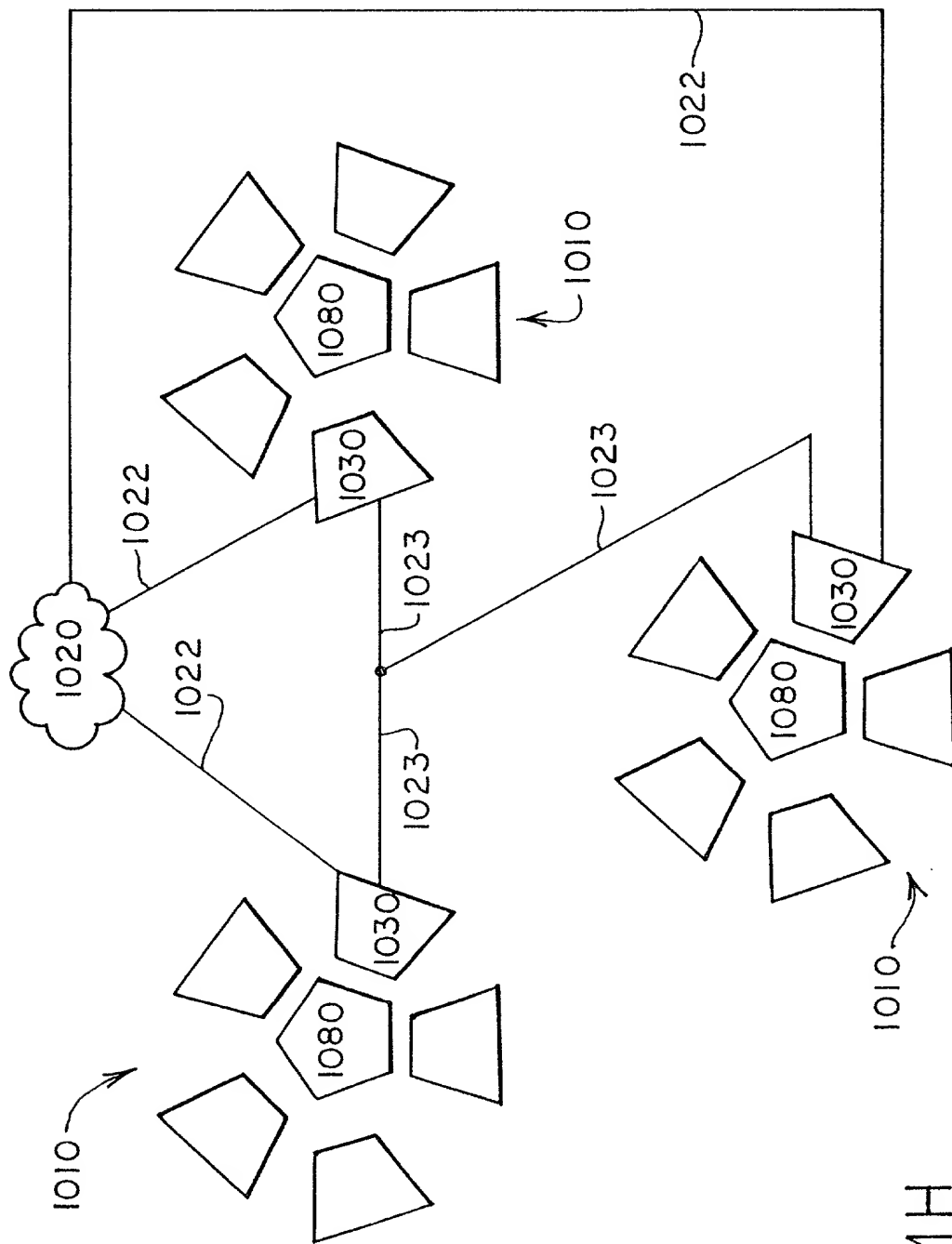


FIG. 1H

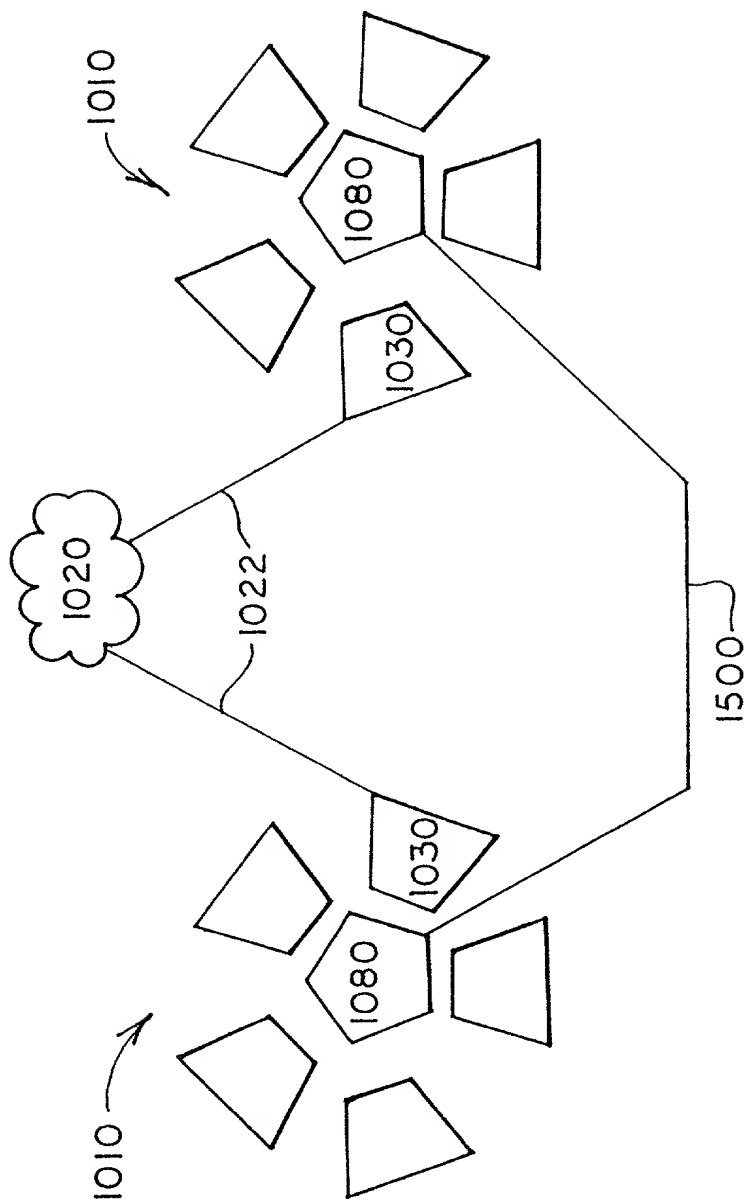


FIG. 11

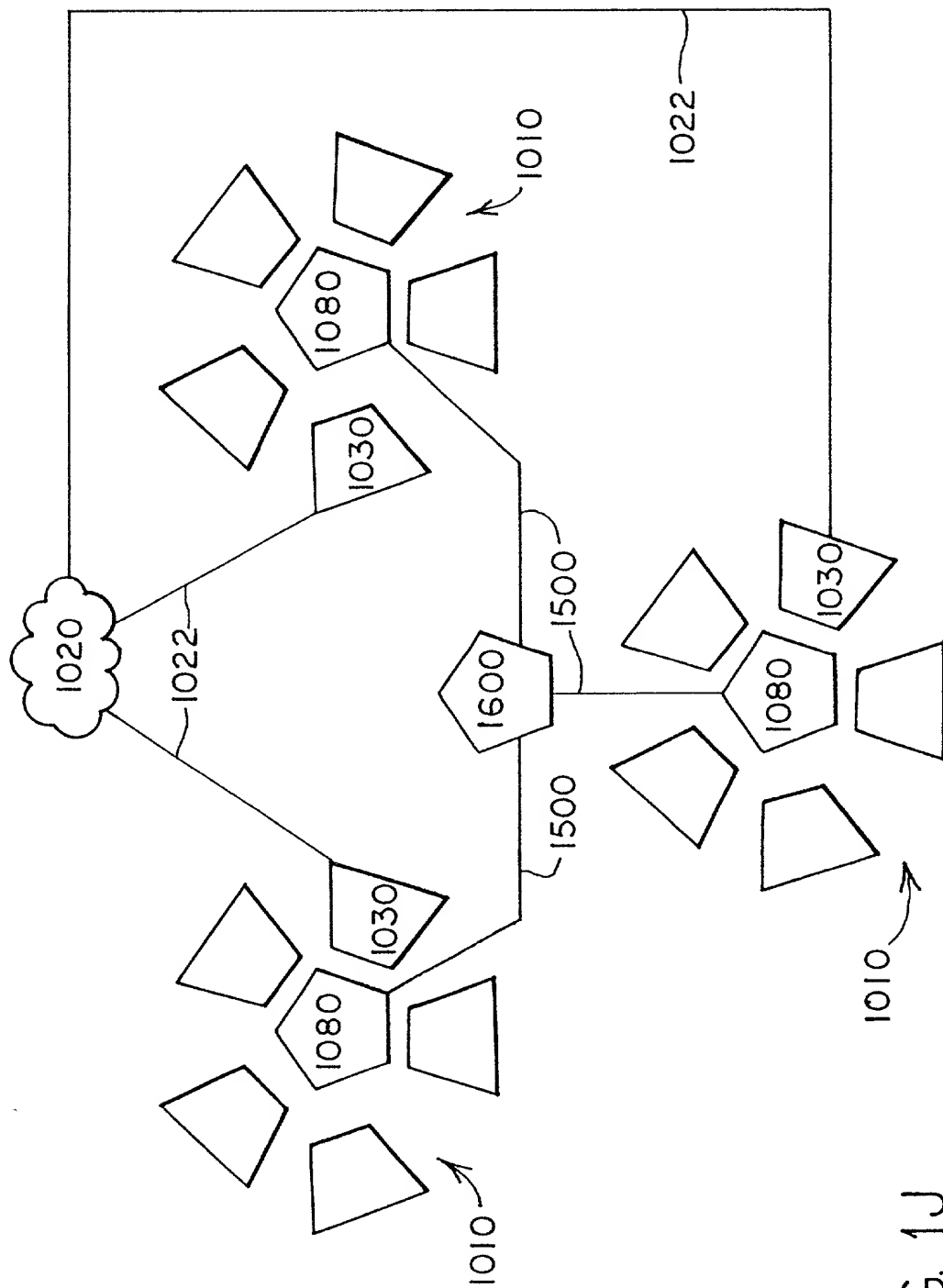


FIG. 1J

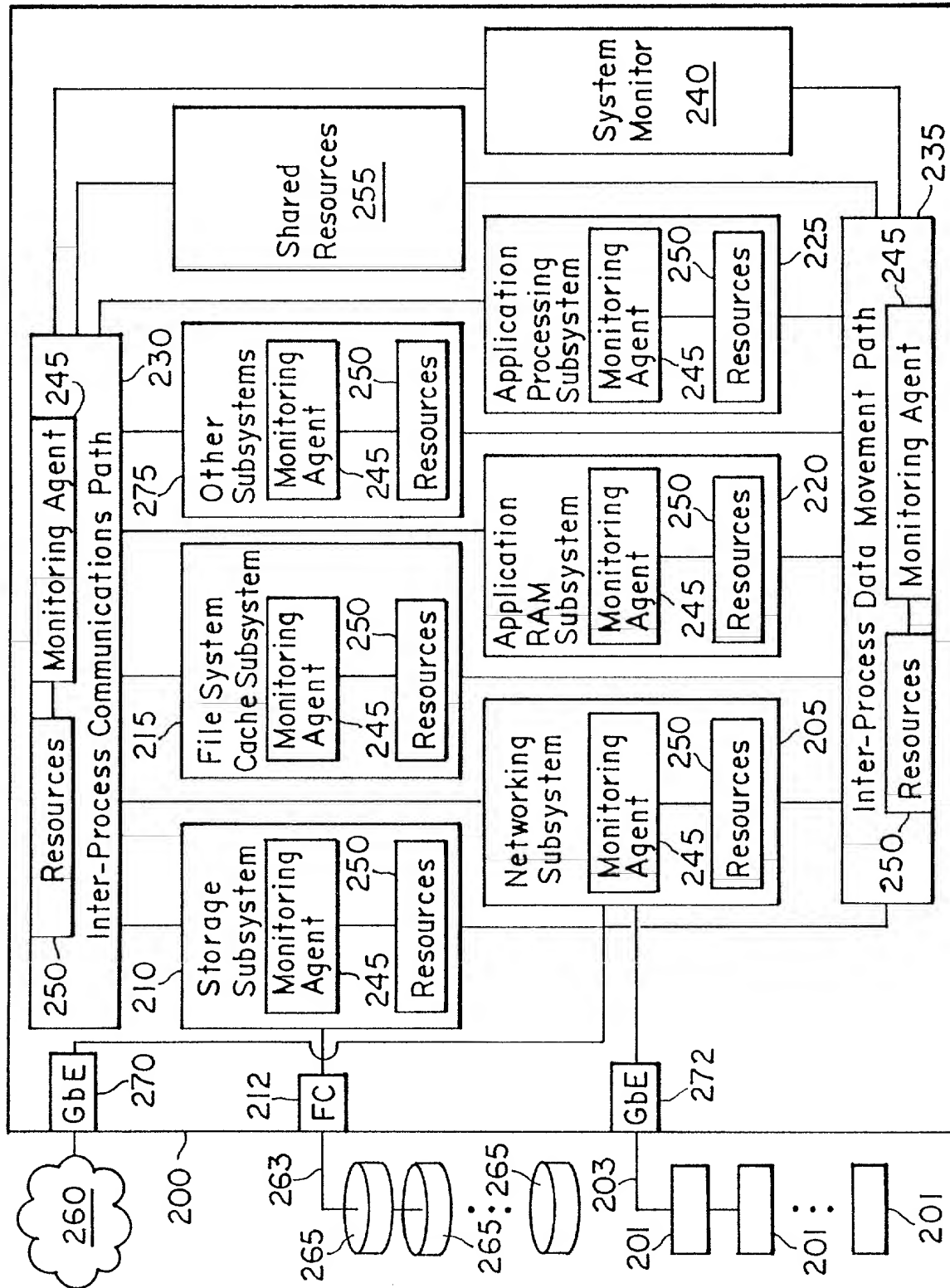


FIG. 2

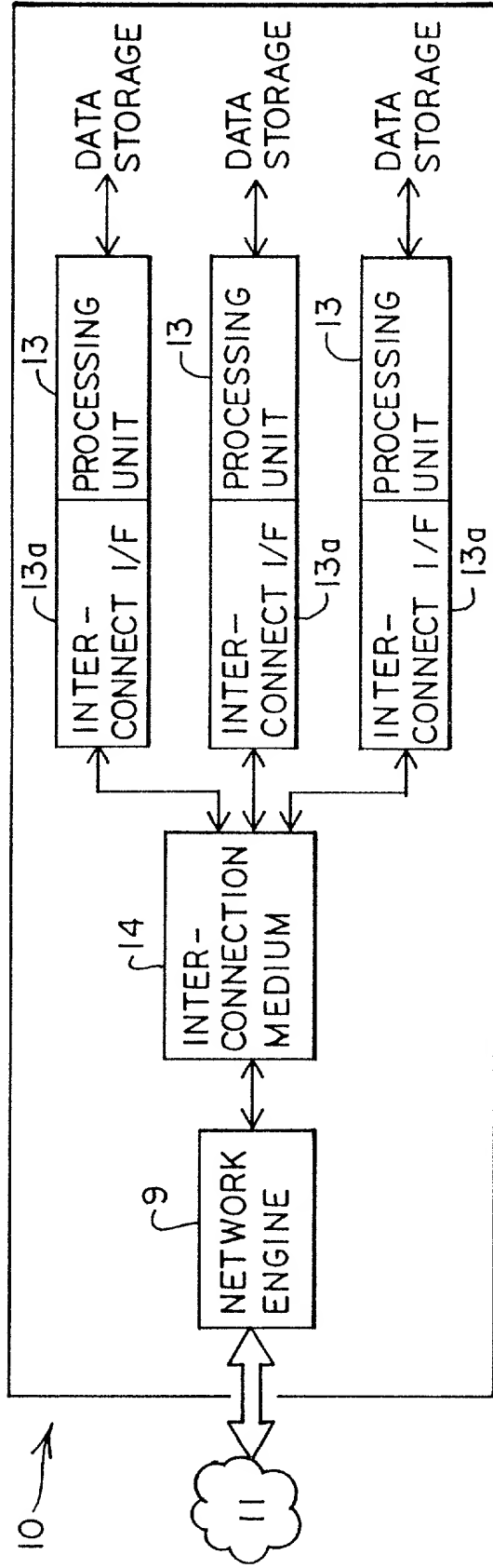


FIG. 2A

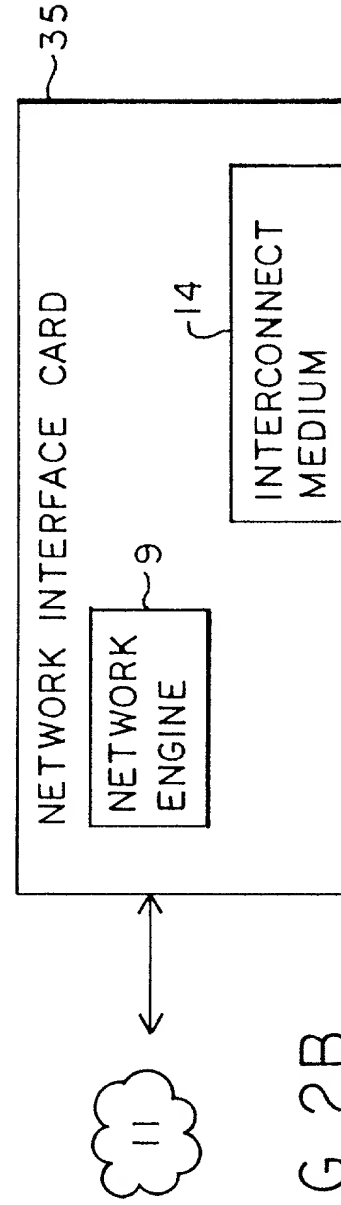
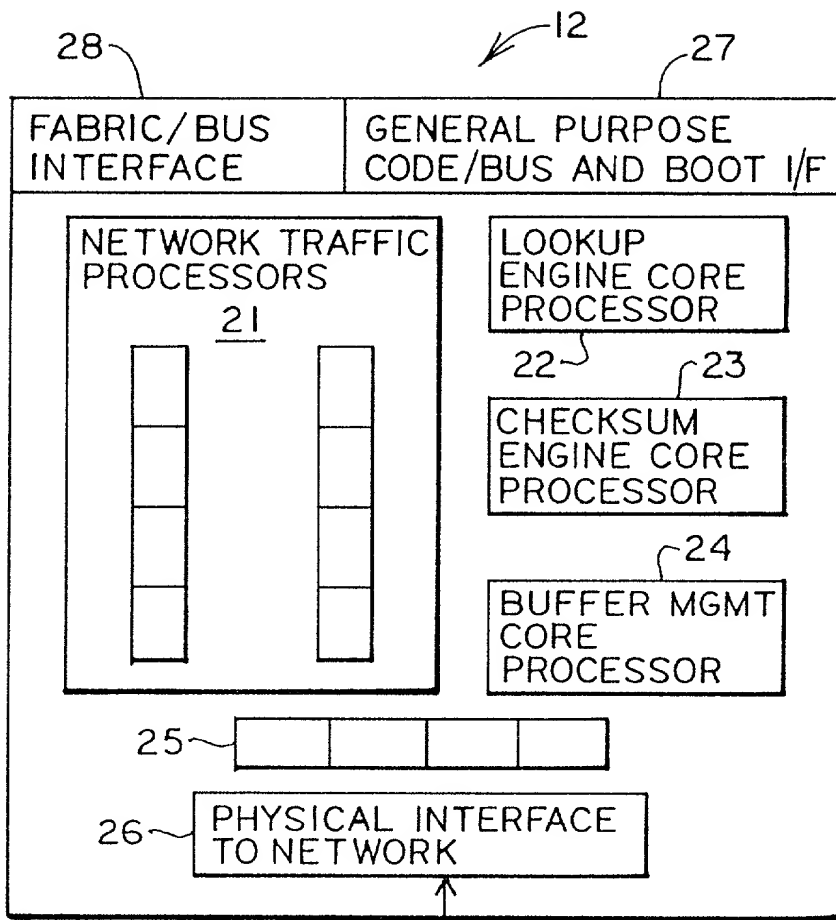


FIG 2B



TO ROUTER

FIG. 3

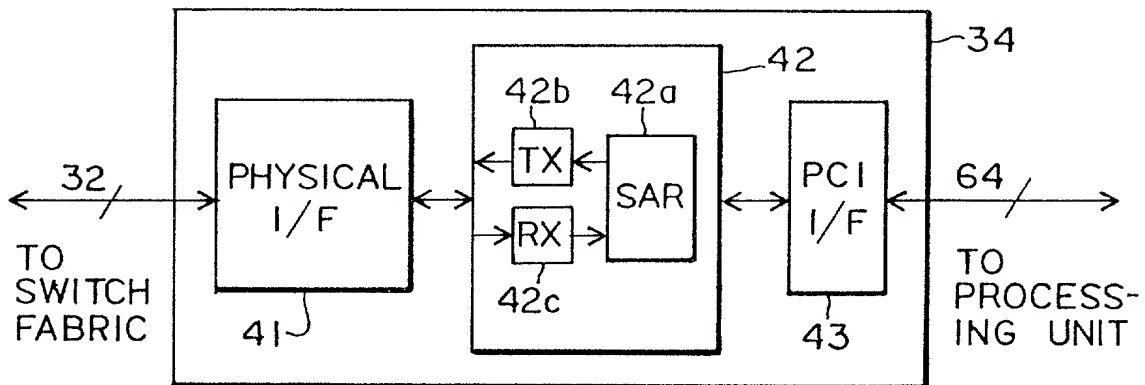


FIG. 4